White Paper Report

Report ID: 104105

Application Number: HD5147511

Project Director: Robert Turknett (turknett@tacc.utexas.edu)

Institution: University of Texas, Austin

Reporting Period: 9/1/2011-8/31/2012

Report Due: 11/30/2012

Date Submitted: 4/9/2013

White Paper Report

Grant Number: HD-51475-11
Project Title: A Thousand Words: Advanced Visualization for the Humanities
Project Director: Rob Turknett
Grantee Institution: Texas Advanced Computing Center (TACC)
Date: November 30, 2012

Definitions

Tiled display/scalable display/gigapixel display/display cluster/visualization wall: A large-scale display made from many desktop monitors, HDTV's, or video projectors, usually driven by multiple computers networked, or "clustered," together. The tiling of multiple displays creates a total display resolution orders of magnitude larger than a single monitor, enabling the viewing of vast amounts of information at once by multiple viewers.

Processing: A computer programming language for creating interactive visualizations, originally designed at MIT's Media Lab as a pedagogical tool to teach programming concepts to media artists. Processing is now used by tens of thousands of new media and visual design professionals and is steadily gaining popularity in the field of information visualization.

Library/Software library/API: A pre-built collection of programming code tailored to a specific purpose. A library can dramatically reduce the amount of time it takes to develop software for a specific purpose, because most of the work is abstracted and automated. The custom part of your code communicates with the software library through the application programming interface (API). The advantage of building a software library versus a software application is that it encourages innovation -- other software developers around the world can take the library and build new things with it.

Project Overview

Visualization uses computers to find patterns and make connections that normally cannot be seen. The Thousand Words project is an initiative to develop software tools to allow humanities researchers to use visualization - specifically on high-resolution displays powered by supercomputers – to perform novel research.

The tools that are currently available for visualization on high-resolution displays are primarily targeted at scientists, and because of that, the tools can be complicated and ill suited to the needs of humanities scholars. To address this issue, we chose a programming language called Processing as a platform on which to build visualizations and visualization tools in collaboration with humanities scholars.

Processing makes it easy to create simple interactive "sketches" that combine text, video, 3D graphics, animation, sound, and more. However, like most software, Processing was previously incapable of running on ultra-high-resolution display clusters. The results of this start up project enable researchers anywhere to use Processing in conjunction with tiled displays at universities, museums and research centers.

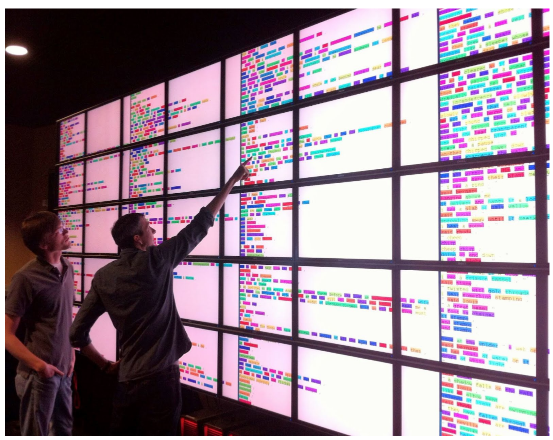
Our long-term goal with this project is to create the world's most advanced platform for humanities visualization. We aim to create software tools that will enable a new class of scholars from the humanities to use high-resolution displays and advanced computing to create visualizations, interactive maps, and multimedia works at a scale and resolution never before possible.

Project Activities

Technical investigations and prototyping

From September 1, 2011 to March 31, 2012, the project team for the Thousand Words project focused primarily on the development of a prototype Processing module to enable interactive visualizations on high-resolution display walls.

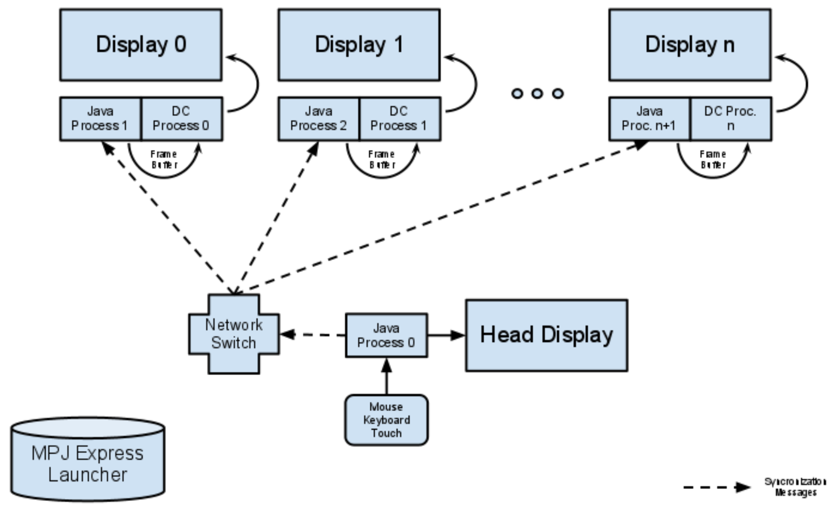
We achieved one of our main goals for the project: enabling Processing sketches to be viewed on tiled displays. Our technical lead, Brandt Westing, is on the development team for a related Texas Advanced Computing Center (TACC) project called DisplayCluster, a new software environment for interactively driving large-scale tiled displays. By leveraging DisplayCluster's pixel streaming capabilities, we were able to view multiple interactive Processing sketches on our 307 megapixel display wall, Stallion. Processing sketches can be viewed by streaming the portion of the user's desktop that contains the Processing sketch. DisplayCluster is open source and has been made freely available to the public.[1]



A Processing sketch, ProseVis, streamed to the tiled display system

While streaming desktops via DisplayCluster satisfies our goal of enabling Processing sketches to be viewed on tiled displays, the streaming resolution is limited to around 2560x1600 pixels (4 megapixels). This method cannot support the native resolution of large displays, which can reach up to 38000x8000 pixels (328 megapixels). Achieving larger resolutions for interactive visualizations requires the solution as outlined in the project proposal. In this scheme, a Processing module distributes rendering to each node in the tiled display, as shown in the diagram below:

[1] http://www.tacc.utexas.edu/tacc-projects/displaycluster



A proposed architecture for a distributed rendering Processing module

Each node runs a Java process that receives messages from the *Processing* module and sends display instructions to the DisplayCluster process running on that node. In other words, the interactive visualization is broken down into sections so that each node in the tiled display computes only the portion of the image that it needs to show.

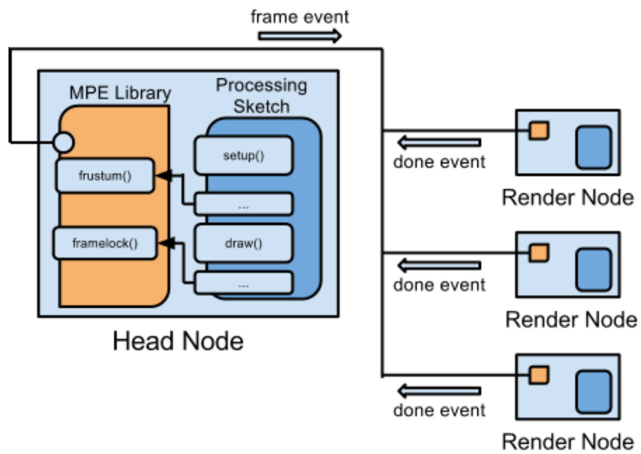
Initially, we planned to use an existing application programming interface called VRJuggler to implement this design. However, technical investigations and experiments revealed that VRJuggler is a dead end. The library is not be easily adaptable to the Java framework in which Processing is based. This was a bit of a setback, since we counted on leveraging VRJuggler to speed development time.

In light of this setback, we focused development on an in-house solution to distributed graphics contexts required by parallel Processing sketches. While researching this idea, we discovered an existing open-source framework called Most Pixels Ever (MPE). MPE is a library for distributed Processing, and appeared from our investigations to be a usable solution to interactive Processing across the tiled display. Rather than re-invent the wheel, we resolved to adapt this library to our needs, and continue the development of the library to make it more general and easily configured for end users.

While MPE provides a good foundation for us to build upon, the design of the framework significantly alters the ease-of-use of Processing, which limits the advantages that Processing has over other graphics frameworks such as OpenGL, VTK, and OpenFrameworks. By using MPE, it is necessary to change the source code of the original sketch significantly to work on a large-scale display system. This presents an obstacle to non-technical users. Our work in extending MPE therefore focused on minimizing these intrusive elements and code modification, and simplifying the creation and maintenance of the configuration files needed to describe the display surface used by the software. Ultimately, due to the large amount of changes required, we decided to develop our own library inspired by MPE called Massive Pixel Environment (originally named Most Pixels Ever: Cluster Edition)[2], rewritten from the ground up.

[2] We recently decided to change the name of our library from Most Pixels Ever: Cluster Edition to Massive Pixel

Development of Massive Pixel Environment



The finalized architecture for Massive Pixel Environment

From March 31, 2012 to September 1, 2012, the project team focused on the development of Massive Pixel Environment. This rewrite of Most Pixels Ever greatly simplifies the process of preparing a Processing sketch for the tiled display, compared with the original version. The toolkit requires only 3-5 lines of code be added to a developer's visualization, and is therefore relatively easy for a developer to extend the visualization from a laptop or workstation to a wall-sized display.

Testing and feedback from collaborators

Throughout the project, the development team for Massive Pixel Environment (referred to henceforth as MPE) worked closely with several collaborators who were interested in using MPE in their research. These initial users were instrumental in gathering requirements and in providing feedback on early versions of the software. These collaborators include Dr. Jason Baldridge (Department of Linguistics, UT Austin), Tom Benton (College of Education, UT Austin), Dr. Tanya Clement (School of Information, UT Austin), Dr. Matt Cohen (Department of English, UT Austin), Dr. Yusheng Feng (SiVert Center, UT San Antonio), and Dr. Craig Tweedie (Cybershare Center, UT El Paso).

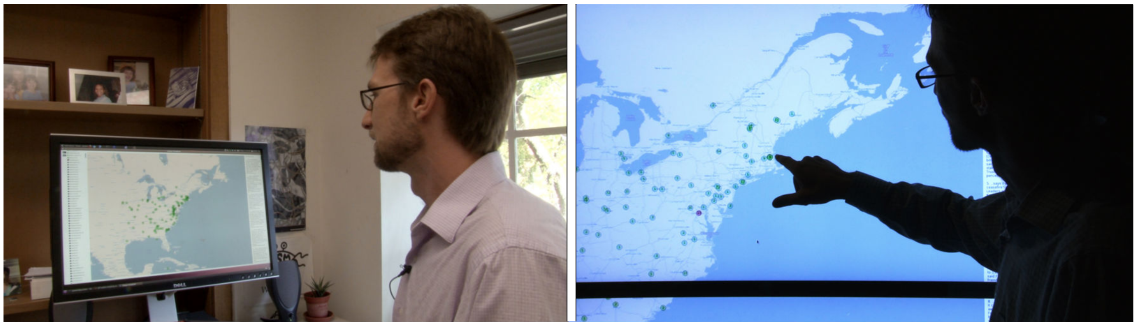
Several collaborators developed Processing-based visualizations in parallel with MPE development, so that when MPE was ready to test, their visualizations could be moved from their laptops to the high-resolution tiled displays in the TACC/ACES Visualization Lab.

Project funding was not used for the development of these visualizations. Instead, we provided technical assistance and early access to the MPE software to our collaborators in exchange for their feedback. Funding

---

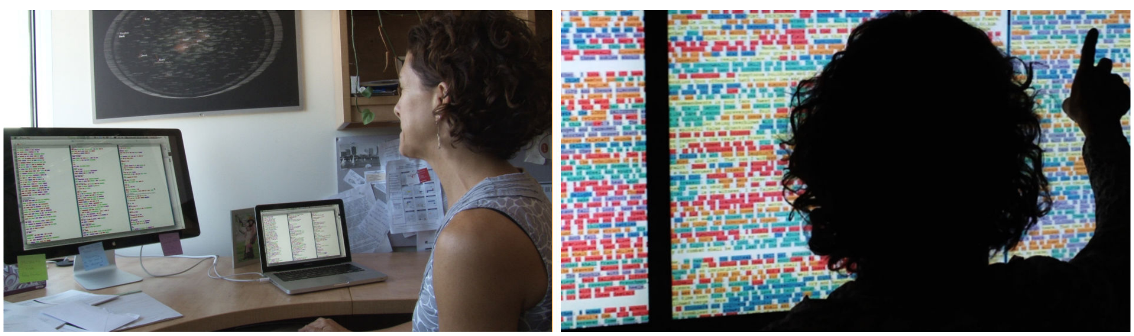
Environment, in order to avoid confusion with Shiffman's original library. We decided to preserve the acronym for continuity. We are currently in the process of changing the name in the codebase, on the web, and in videos.

for the work on these visualizations came from a variety of sources, including Mellon, the National Science Foundation (NSF) and UT Austin.



Dr. Jason Baldridge interacting with a visualization of Civil War archives in his office and at the TACC/ACES Vislab

Dr. Jason Baldridge at the University of Texas at Austin Department of Linguistics and graduate research assistant Mike Speriosu developed a large-scale interactive map showing places mentioned in a large archive of Civil War documents. Clicking on a marker shows the name of the document, the page number, and an excerpt of the place mentioned. The high-resolution display allows much more of the data to be visible at once, giving researchers a new and more comprehensive geospatial view of the archive.

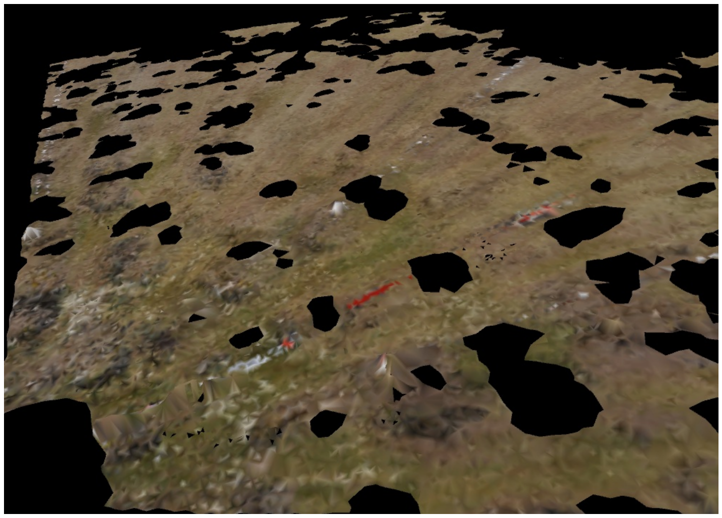


Dr. Tanya Clement, using ProseVis in her office and at the TACC/ACES Vislab

Dr. Tanya Clement at the University of Texas at Austin School of Information and undergraduate research assistant Christopher Wiley worked with the Thousand Words team to develop a new version of the ProseVis language visualization tool, suited for use with high-resolution displays. This version enables the display of many more documents and many more passages for cross comparison.

We also had several early users who tested MPE at other visualization labs within the UT system. Dr. Craig Tweedie at the University of Texas at El Paso's Systems Ecology Lab used MPE to visualize LIDAR datasets of the Antarctic Tundra. Dr. Tweedie's research involves high resolution aerial point clouds, and MPE was used at the University of El Paso's Center for Cybershare's Amythest tiled display to view the

LIDAR point clouds at large scale.



A Processing visualization of Antarctic ice sheets generated from LIDAR data

Dr. Yusheng Feng at the University of Texas at San Antonio participated in early usage of MPE at the SiVirt Visualization Laboratory's Nemo Tiled Display. Dr. Feng's group used MPE to produce high-resolution demonstrations and visualizations for visitors and researchers related to scientific visualization. MPE is one of two primary display libraries currently in use at the SiVirt VizLab.

Software release and publicity

The software was released to the general public on December 11, 2013 via the open-source project hosting website, GitHub. We had planned to release the software in September, but a hardware upgrade of TACC's tiled display, Stallion, took longer than anticipated and delayed our work significantly.

The Texas Advanced Computing Center published a press release and video to announce the software and build interest, and promoted the release to its followers through social media, including Facebook and Twitter. The announcements were virally shared, which led to new connections and collaborations from interested researchers at other institutions, including Daniel Shiffman, developer of the original Most Pixels Ever library, Casey Reas, one of the original creators of the Processing language, and Lev Manovich, who began collaborating with us on our proposal to continue the project.

We also promoted the software to the Processing community through the highly trafficked user forums at processing.org. The libraries page at processing.org will soon contain a link to the MPE software page.

Accomplishments

With the release of Massive Pixel Environment, we achieved our main goal of developing an open-source software library that enables Processing to work with high-resolution tiled displays. By leveraging Processing, MPE provides a rapid prototyping platform to develop large-scale, high-resolution interactive visualizations for tiled displays. This software makes developing interactive visualizations for tiled displays far easier and less time-consuming, and significantly lowers the barrier to entry for researchers in the humanities.

Interest from the open source community will be crucial for the future development of MPE. As of this writing, the MPE source code has been "starred" 22 times and "forked" 5 times on GitHub. A star means that someone in the GitHub programming community has added the project to their list of favorites. A fork means that a member of the community has created a separate project using the MPE source code as a starting point. This indicates a good level of initial interest, and we expect it to grow as we continue to promote the MPE library. It is currently the most active software project in TACC's GitHub repository.

GitHub does not allow us to track downloads of the library .zip file, so we have moved this file to SourceForge, which will allow us to track downloads in the future.

Evaluation

Overall, the Thousand Words project was very successful. We met our main goal of developing a platform to create interactive visualizations on high-resolution tiled displays that is much easier to use than anything that existed previously. Feedback from early usage internally and with our collaborators has shown that the MPE platform greatly simplifies and speeds the development of interactive visualizations for many-monitor displays.

While the project met its goals, we faced a number of challenges along the way:

Early work on the project revealed our proposed design to be flawed (as described in the Project Activities section of this report).

This is quite common in software development – it is often difficult to anticipate technical issues at the outset of a project, without actually doing the work. For the same reason, time estimates can change dramatically. We dealt with this by finding another technical solution to accomplish our goal. However, it is easy to imagine a scenario in which we would have had to scrap the original idea and change course completely, because the idea was no longer technically possible or no longer feasible within the proposed budget.

We were unable to find a way to fund a visualization collaboration with Professor Matt Cohen in the UT Department of English.

We pursued an idea to create a visualization of the WorldCat database, but Dr. Cohen did not have funding for a student to work on the project, and we did not have any extra funds to support the work. Our other collaborators were already working on projects involving data visualization, which lowered the funding threshold and provided mutual benefits for all involved.

The lack of basic computer programming expertise in the humanities made it difficult to involve humanities students on the project.

It is interesting to note that both of our collaborators who produced humanities-related visualizations are from non-humanities departments, and the students who did the programming were from the department of computer sciences and (computational) linguistics. We believe that more institutional opportunities are needed for students in the humanities to learn computer programming, and to work together with students in computer science on collaborative digital humanities projects (which would provide mutual benefit).

We found it difficult to port a traditional menu-based application to Processing.

In our attempt to port ProseVis from the desktop to the tiled-display, we discovered that the desktop graphical user interface was very difficult to implement in Processing. The Java Swing GUI framework is not available in Processing, and the language is not intended to build these kinds of applications. While Processing is great for creating interactive visualization sketches, it is not intended or well-suited to create traditional desktop-style applications. Furthermore, user interfaces designed for a desktop screen often do not work well on large, high resolution display.

- **Best Practice:** Use Processing for interactive visualization sketches, not for developing Windows or Mac-like applications with lots of menus.
- **Best Practice:** Design interactive visualizations with a large, high resolution display in mind. Do not assume a desktop application will translate well to a high-resolution display.

Bezels on tiled displays caused issues for reading text.

On tiled displays, there is dead space in between the bezels. Typically, the display is configured so that the bezels obscure the part of the image that is "behind" the bezels (creating a windowpane effect). Any portion of the text that is behind the bezels is obscured, making it less than ideal for text display. As monitor bezels get smaller, this will hopefully become less of an issue.

- **Best Practice:** Consider whether bezels will cause a problem when designing visualizations for a tiled-display.

Continuation of the Project

Due to the interest in MPE from arts, humanities, and sciences, as well as its usefulness to TACC's own visualization group, we intend to continue development of MPE. We also intend to continue publishing results from the project, and plan provide training through TACC's user services program[3].

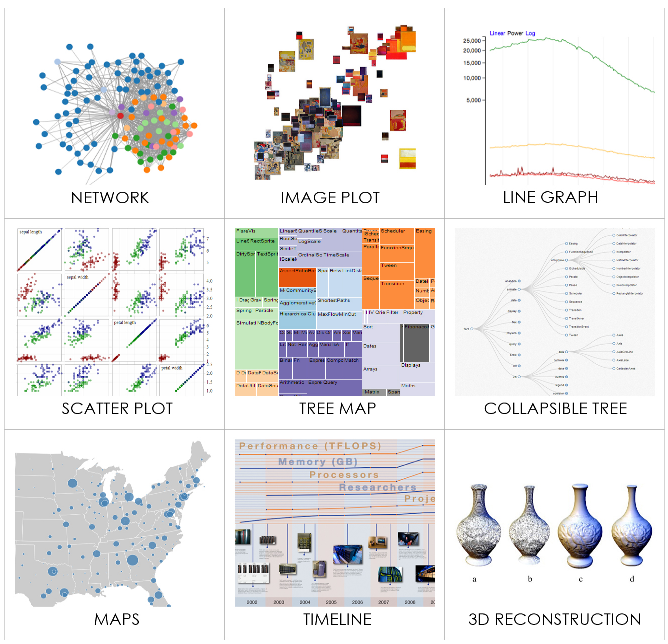
In addition, a number of new collaborative partnerships were formed as a result of the Thousand Words project that will continue after the grant period. We continue to collaborate with all of our early MPE users both formally and informally. For example, TACC is providing high performance computing and visualization resources for the High Performance Sound Technologies for Access and Scholarship (HiPSTAS) Institute in May 2013, organized by Dr. Tanya Clement.[4] The Institute will include an instructional presentation of ProseVis and MPE at the TACC/ACES Vislab. The software release also led to a promising collaboration with cultural analytics researcher Lev Manovich, Professor at The Graduate Center, CUNY.

TACC is currently seeking funding to extend the Thousand Words project by executing along two parallel thrusts of software development. First, we intend to add additional enhancements to MPE in response to current and future user feedback. Currently planned enhancements will make MPE even easier for humanities scholars to install and use on a scalable display system. Since these systems are often managed by people

[3] http://www.tacc.utexas.edu/user-services/training
[4] http://blogs.ischool.utexas.edu/hipstas/2012/11/14/welcome-to-hipstas/

outside a scholar's home department, we would like to make the process to get up and running as simple as possible. These enhancements will make moving back and forth from laptop to visualization wall a seamless process. Other planned enhancements will add support for gestures, touch, and wearable displays, and meet additional needs identified by our user base.

Second, we plan to develop a 1000 Words software library for Processing that allows programmers to quickly build visualizations relevant to the humanities. This library will consist of pre-made algorithms and visual elements for creating various types of information visualizations, along with examples and sample data sets specific to the humanities to provide a launching point for other scholars. Working with our faculty collaborators, we have identified a preliminary list of visualization types that we intend to support, including networks, line graphs, scatterplots, image plots, trees, maps, timelines, and 3D point clouds. Examples will include visualizations such as citation networks, document similarity networks, imageplots of paintings, scatterplots of poems by linguistic features, graphs of Google N-gram data, 3D reconstructions of scanned historical artifacts, and so on.

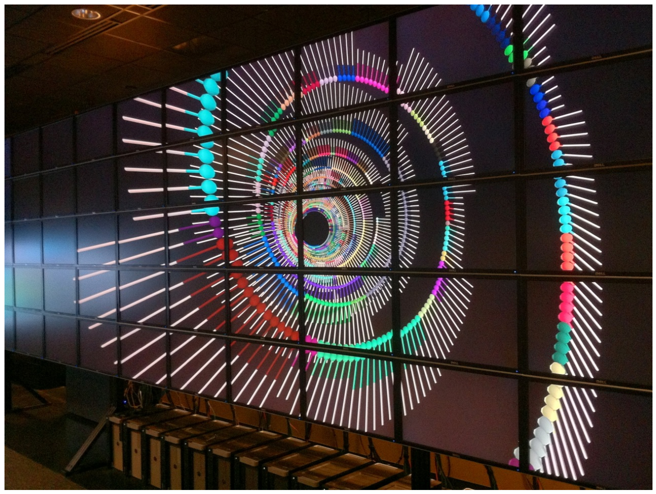


Examples of visualization types that the 1000 Words library will support

By creating MPE and the 1000 Words library, we hope to build a bridge between the visual art and design community, the vibrant open source community of Processing artists and coders, the information visualization community, and the community of scholars in the humanities.

Long Term Impact

MPE has become a central tool for TACC's visualization group. MPE is also in use at several other university visualization labs, and we expect it to reach more labs over the next year as we continue to promote and improve it. Interest from the scientific visualization community, the media arts community, the open source community, and other visualization labs will help ensure MPE's continued development, and opens up a range of potential funding opportunities. TACC's visualization group has recently submitted a proposal to NSF's Human Centered Computing (HCC) program which, if awarded, will fund additional applications and development of MPE.



An interactive visualization of high performance computing queues using MPE

We believe that MPE holds great promise for creating interactive visualizations for museums, libraries, and other collections-based institutions. If we are able to secure funding to produce a 1000 Words visualization library full of humanities-related examples, we will be able to demonstrate its potential and create even more interest within the humanities.

Media artists, creative coders, and humanities researchers are just beginning to scratch the surface of what can be created with MPE, but we believe that the Thousand Words project has put TACC well on its way to building the world's most advanced platform for humanities visualization. With continued funding and development, we believe that the Thousand Words project can meet this lofty goal.

Grant Products

A web page for the project was created at http://www.tacc.utexas.edu/tacc-projects/a-thousand-words during the early stages of the project and is continually updated with current information. TACC also maintains a web page for the MPE software, located at http://www.tacc.utexas.edu/tacc-software/massive-pixel-environment.

We produced two videos about the project. The first, *A Thousand Words: Advanced Visualization for the Humanities*, was published to TACC's YouTube page on December 12, 2012, and has received 1,019 views to date:
http://www.youtube.com/watch?feature=player_embedded&v=kvOuJ2RwBTA

The second video, *Massive Pixel Environment: A Tool for Rapid Prototyping with Distributed Displays*, has not yet been promoted due to the pending name change:
http://www.youtube.com/watch?feature=player_embedded&v=a0alT390XaI

The source code for the project is available through TACC's GitHub account, at https://github.com/TACC/MassivePixelEnvironment.

Documentation and tutorials on how to install and use the MPE Processing library are located on the project GitHub page. This documentation is continually updated as we make changes to the library.

Listed below are the conference submissions about the project to date (all accepted):

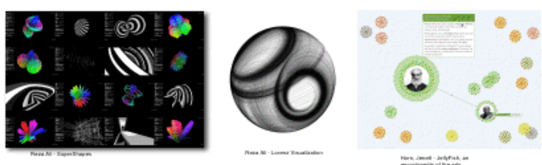
- Westing, B., Nieto, H., Turknett, R., Gaither, K. MostPixelsEverCE: A Tool for Rapid Development with Distributed Displays, CHI 2013 Extended Abstracts, April 27–May 2, 2013, Paris, France.
- Westing, B., Turknett, R. Extending the Processing Programming Environment to Tiled Displays. IEEE Visualization, 2012.

Appendix 1: IEEE Visualization 2013 Conference Poster

Extending the Processing Programming Environment to Tiled Displays

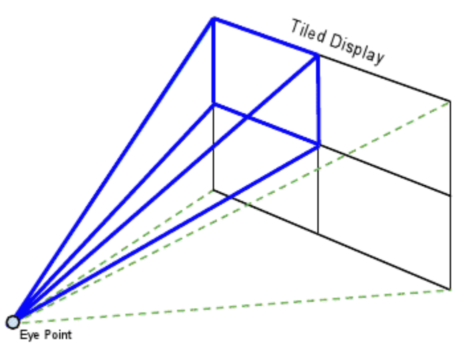
Brandt Westing (bwesting@tacc.utexas.edu)
Robert Turknett (turknett@tacc.utexas.edu)
The University of Texas At Austin
TACC *Visualization Laboratory*

Processing is an open source programming language and environment for people who want to create images, animations, and interactions. Initially developed to serve as a software sketchbook and to teach fundamentals of computer programming within a visual context, Processing also has evolved into a tool for generating finished professional work. Today, there are tens of thousands of students, artists, designers, researchers, and hobbyists who use Processing for learning, prototyping, and production. -Processing.org

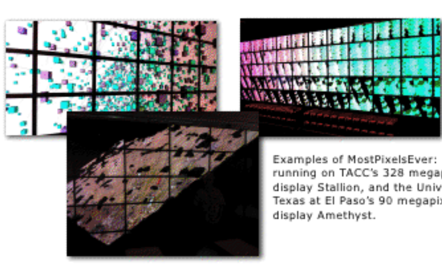


MostPixelsEver – Cluster Edition is an extension of the Processing programming environment that enables visualization in cluster-driven display environments without extensive knowledge of programming languages, graphics interfaces, or distributed computing. *MostPixelsEver:CE* is heavily inspired by Daniel Shiffman's work in [4], building upon it with a more efficient and configurable implementation. The work described here enables visual artists, humanities scholars and students, and even traditional programmers to create interactive visualizations in high-resolution distributed environments with simplicity. *MostPixelsEver:CE* hides the inherent complexity of distributed environments by abstraction, and makes it possible to rapidly create visualizations on large displays.

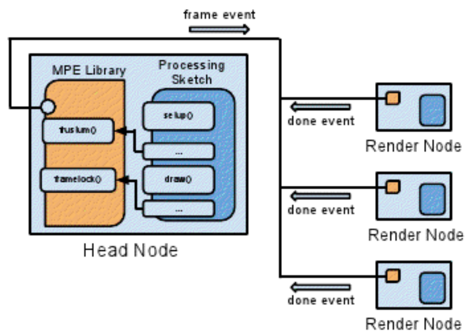
Most tiled-display consist of several monitors or display media, driven by a cluster of computer that render the content needed. This solution does not facilitate graphics workflows, however, as each computer in the cluster maintains its own context separate of the others. *MostPixelsEver:CE* unifies the context between the separate machines in the cluster and gives the entire composite surface a single point of view.



Before the scene is drawn, the main instance transparently calls the messaging *Process* object to adjust the **view frustum** such that the scene appears correct within the larger display surface. This call ensures that the entire scene is not drawn to the local window, but that only the correct portion of the scene is drawn with respect to the larger display area.

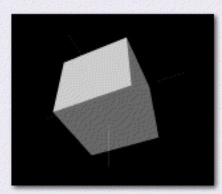


Examples of MostPixelsEver: CE running on TACC's 328 megapixel display Stallion, and the University of Texas at El Paso's 90 megapixel display Amethyst.



After the scene has been drawn, but before the scene is pushed to the graphics pipeline for rasterization, a post-draw method is called on the *Process* and causes the Processing instance to halt until the frame lock has been unlocked. The frame lock is controlled by the Process instance and unlocks the frame lock when a frame event message is sent from the master messaging *Process* (or head node). This sequence of events effectively synchronizes the display surface such that each process must be ready to render the scene before the scene is rendered. While this has the negative side effect that the slowest process controls the speed of rendering, it guarantees that the display is in sync.

```
Process process; // MPE Process thread
Configuration tileConfig; // MPE Configuration object

void setup() {
  // create a new configuration object
  tileConfig = new Configuration("configuration.xml", this);

  // set the size of the sketch based on the configuration file
  size(tileConfig.getLWidth(), tileConfig.getLHeight(), OPENGL);

  // create a new process
  process = new Process(tileConfig);

  // start the MPE process
  process.start();
}

void draw() {
 rotateY(-.5);
 background(0);
 fill(255,0,0);
 box(200);
}
```



Acknowledgments

This work was made possible by funding from the National Endowment for the Humanities (NEH) Grant: HD-51475-11, A Thousand Words: Advanced Visualization for the Humanities.

References

[1] K. Doerr, F. Kuester, "CGLX: A Scalable, High-performance Visualization Framework for Networked Display Environments," *IEEE Transactions on Visualization and Computer Graphics*, vol. 99, 2010.

[2] S. Eilemann, M. Makhinya, R. Pajarola, "Equalizer: A Scalable Parallel Rendering Framework," *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 3, 2009.

[3] *Processing*, Cover, B. Fry, C. Reas, date last referenced: 06/25/2012, http://processing.org/.

[4] *Daniel Shiffman*, Most Pixels Ever, D. Shiffman, date last referenced: 06/25/2012, http://www.shiffman.net/2007/03/02/most- pixels-ever/.

Appendix 2: CHI 2013 Conference Paper

MostPixelsEverCE: A Tool for Rapid Development with Distributed Displays

Brandt Westing
TACC Visualization Lab
201 E 24th St
Austin, TX 78712 USA
bwesting@tacc.utexas.edu

Heriberto Nieto
TACC Visualization Lab
201 E 24th St
Austin, TX 78712 USA
hnieto@tacc.utexas.edu

Robert Turknett
TACC Visualization Lab
201 E 24th St
Austin, TX 78712 USA
turknett@tacc.utexas.edu

Kelly Gaither
TACC Visualization Lab
201 E 24th St
Austin, TX 78712 USA
kelly@tacc.utexas.edu

Abstract
We describe a software library called MostPixelsEver: Cluster Edition (MPE) for use in visualization, arts, humanities, and interface prototyping in distributed display environments. We discuss the implementation of the software and its unique qualities when contrasted with other distributed graphics libraries and environments in the areas of interaction, rapid development, and rich library support. We provide concrete examples of its usage at multiple sites, lessons learned, and a discussion on the future of tiled display environments.

Author Keywords
distributed graphics; tiled displays; visualization; interface prototyping; arts; humanities

ACM Classification Keywords
I.3.2 [Graphics Systems]: Distributed/network graphics

General Terms
Design; Human Factors

Introduction
Software development with tiled displays can become cumbersome when the number of compute resources required to drive the display(s) is greater than one. It is necessary for a cluster, or multiple compute resources, to

drive a display system when a single computer lacks the necessary graphics or computation capabilities to drive the display(s) alone. These systems are difficult to use because of the lack of general purpose software developed for distributed systems. Several libraries and algorithms have been developed to ease the burden of graphical application programming for such systems. However, we have found these frameworks cumbersome, invasive, or unsatisfactory for novice users. Past developments have proved to be particularly difficult for non-computer scientists to use because of the knowledge necessary to navigate low-level programming interfaces.

MPE was developed to solve problems that are seen often with distributed display systems: How can users be enabled to quickly create visualizations, researchers to easily test interface prototypes for interaction studies, and non-computer scientists the functional literacy to utilize distributed displays effectively?

Background

Distributed display systems come in many shapes and sizes. Seamless displays can be created from multiple projectors driven by multiple compute resources. Majumder[10] describes the practical design of multi-projector systems. Tiled Liquid Crystal Displays (LCD) can be used to create high resolution and high aggregate pixel count displays suitable for high resolution visualization. Countless tiled LCD displays exist, the largest of which include the RealityDeck[12], Stallion[11], and the HIPERWall OptiPortal[3].

The development of software for these display systems has been as varied as the systems themselves. Three common architectures have been employed to simplify the usage of distributed displays: pixel streaming, sort first non-invasive, and sort first invasive rendering. Pixel streaming architectures perform computation and rendering at one or more compute resources and stream the rendered pixels to rendering resources that then display the pixel stream images. Pixel streaming architectures require high computational ability at the application node (which performs computation and rasterization), and large available bandwidth for the pixel streams - especially for dynamic content. The Scalable Adaptive Graphics Environment (SAGE)[7] is an example of a pixel streaming architecture. Chromium[6], in its *tile-sort* mode, is a sort first non-invasive architecture for distributed rendering. Chromium intercepts the OpenGL stream from the application node(s) and streams the commands to the render nodes, which render a section of the final image shown on the display. Sort first invasive architectures are those that require calls within the application code (invasive) for distributed rendering to function. These architectures have a complete graphics pipeline at each render resource. Representative sort first invasive architectures include cglX[4], DisplayCluster[8][1], Equalizer[5], and MPE (described here). MPE, however, is the only sort first invasive architecture based around a higher-level programming language with ease-of-use and a visualization focus as primary considerations.

Importantly, MPE is inspired in function and namesake by an earlier work by Daniel Shiffman[13] intended for distributed displays with Processing. Unlike Shiffman's previous work, the work presented here provides support for the latest Processing versions, high numbers of distributed hosts, optimized synchronization, and easier configuration through the use of a single configuration file.

[1] In the *parallel streaming* mode. Normal operation displays only static content such as images and video.

The Processing Language
Processing is a programming environment that was developed in 2001 to promote software literacy in the visual arts[1]. It is a free and open-source programming language and development environment that was originally developed to teach fundamentals of computer programming, but quickly developed into a tool for creating professional work. Processing abstracts complicated programming concepts and simplifies the application work flow by: hiding compilation, linking, and running the program executable; by implementing a scripting layer on Java; and by providing a simplified development environment. Furthermore, Processing provides a programming construct broken into two functional components that abstract the control loop found in most applications.

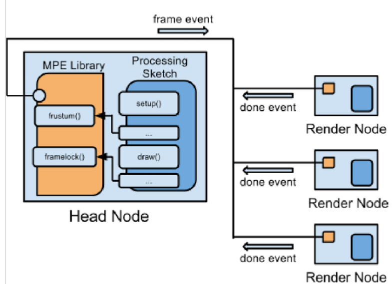


Figure 1: The MPE rendering process uses a centralized barrier synchronization method over TCP.

Implementation

MPE provides an abstraction to the distributed environment and aims to make the jump from serial to parallel programming as simple as possible for the developer. As such, development with MPE differs from developing with a serial Processing environment by only two ways: the configuration description file and a small number of MPE setup calls.

The Configuration File
The configuration file is a single XML file that describes the distributed display system. The file specifies the host machines involved in the distributed system, the displays associated with the hosts, and the resolution and orientation of the displays. The file serves to enable the software to properly set the view frustum and configure synchronization among the rendering clients.

The MPE Setup and Rendering Process
When running a distributed program using MPE and Processing, the user makes an MPE function call within the *Sketch* (main program) setup code that starts a communication thread that runs in parallel to the main thread. The communication thread handles synchronization between the hosts in the distributed system. Specifically, a centralized barrier synchronization method is used over TCP sockets between a leader process and the following or rendering processes. Communication messages are simple 8-bit messages that represent a *frame* event or *done* event. The *frame* events are sent from the central process and allow the rendering client(s) to unlock their frame lock (whereupon the scene is rendered), while the *done* events (sent from the followers) allow the central process to provide a barrier until all rendering clients have rendered. When the leader process receives all the *done* events, it broadcasts a *frame*

event. This looping process can be seen in Figure 1.

Before the frame is rendered by a follower, the view frustum is culled such that the rendered scene reflects only the portion of the display area the display is responsible for. For instance, a machine driving a quadrant of a distributed display (1 screen out of 4), will render only the part of the scene that will be viewable in the quadrant.

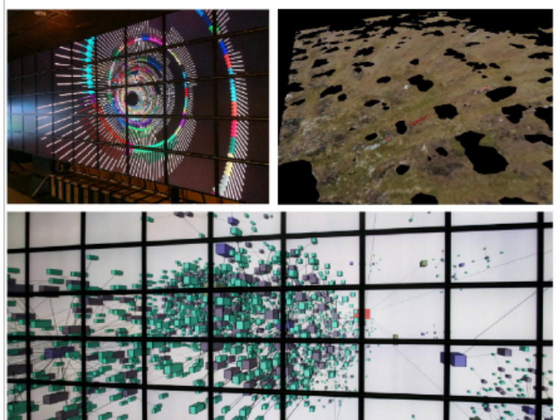


Figure 2: Visualizations using MPE. Top Left: Interactive Visualization of HPC Queues. Top Right: Antarctic Ice Sheet Visualization. Bottom: Structural Visualization of Shakespeare's The Tempest.

Results

We provide three usage examples of MPE in tiled display settings. Furthermore, we contrast the usage of MPE against other distributed display libraries and toolkits.

One: Interactive Visualization of HPC Queues
Using Paul Bourke's visualization of HPC queue statistics as an inspiration[2], the Texas Advanced Computing Center's Ranger Supercomputer queue is visualized in real-time on the Stallion 328 mega-pixel display (shown in Figure 2, top left). MPE is used to facilitate this visualization on the Stallion system, which is composed of 21 distributed hosts. The visualization shows usage of Ranger's 4000 compute nodes, showing 100's of active jobs and features control via multi-touch gestures from a tablet. Multi-touch is enabled through the application of the TUIO[9] protocol to Processing. The resolution of Stallion allows the complete Ranger queue to be visualized and spatially explored.

Two: Scientific Visualization of Antarctic Ice Sheets
Light detection and ranging (LIDAR) data from Antarctica's coastline ice sheets is an important source of information on climate change. The time varying data can reflect changing temperatures on earth. The Systems Ecology Laboratory at the University of Texas at El Paso, along with researchers at TACC, used MPE to visualize LIDAR datasets on the 45-node Amethyst tiled display (Figure 2, top right) at the University of Texas at El Paso. The interactive visualization served to allow researchers to collaboratively and spatially explore the data collected and envision the possibility of comparing large sets of LIDAR data on Amethyst to better understand climate change on earth.

Three: Structural Visualization of The Tempest
Processing is a powerful tool for information visualization and has proven to be useful to researchers who need to quickly visualize the organization of data and/or its structure. This visualization illustrates the structure of text, and is called the *Text Universe*. The *Text Universe* shown here is an excerpt from Shakespeare's *The Tempest*, revealing the structure of Shakespeare's prose using a node graph(Figure 2, bottom). Inspired by a prototype visualization by Tiemen Rapati, this visualization is intended to motivate humanities and arts researchers or students to take advantage of distributed displays as a new medium for exploration, exposition, and insight. The visualization is shown running on TACC's Stallion tiled display.

Discussion
MPE directly compares to the Cross Platform Cluster Graphics Library (cglX): they are both libraries used to develop applications for distributed displays with a sort first invasive architecture. cglX is implemented in C++ and provides an API at the OpenGL level. cglX provides the same type of synchronization as MPE, and operates over TCP sockets: therefore their execution requirements and messaging overhead are similar. clgX may be preferred by a developer who works at a lower level, while MPE will clearly be preferred by those with less experience, less time, or desire the rich libraries that Processing has to offer for visualization, input, and data processing. When compared to pixel streaming architectures, MPE uses far less bandwidth due to its small synchronization messages (compared to pixel content). It is latency sensitive, as are the pixel streaming architectures in SAGE and DisplayCluster. Compared to the sort first non-invasive architecture of Chromium, MPE requires less bandwidth and has similar latency constraints. MPE programs must be modified in the setup phase of a *Sketch*, unlike Chromium which intercepts OpenGL calls from the application and can run application unmodified.

The display architectures mentioned above all have positive and negative attributes. While these libraries and middle-ware have an important place in distributed display systems, there is a current trend in the capability of graphics hardware to drive a greater number of pixels. The adoption of display standards such as DisplayPort allow graphics pipelines to driver higher amounts of pixels across a single output, and hardware vendors are supporting an increasing number of ports per graphics card. It is possible now to see display systems driven by a single computer that drive a large number of displays. In these systems, there is no need for distributed display middle-ware and native applications may run unimpeded (clearly a winning scenario for developers who do not wish to modify code-bases). It remains to be seen how these systems will evolve and whether distributed display architectures (as mentioned here) will be needed in the future to drive tiled displays. It is certain, however, that distributed display software will be an important part of the future of display environments: wearable displays and ubiquitous display environments both require the notion of distributed display architectures if they are to function as a community of devices.

Conclusion
MostPixelsEver: CE (MPE) is a tool for rapid development with distributed display systems. MPE abstracts the parallelism of such an environment from the user and allows for the creation of visualizations, deployment of interface prototypes, and non-expert use of systems that are inherently challenging to develop for. The architecture of MPE is scalable: MPE has been

Appendix 3: Project Web Pages

TACC TEXAS ADVANCED COMPUTING CENTER

Powering Discoveries That Change The World

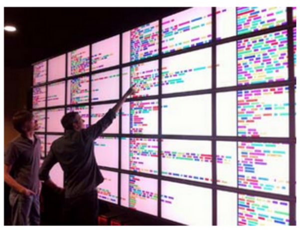
THE UNIVERSITY OF TEXAS AT AUSTIN

Home About Resources User Services Research & Development Partnerships Education & Outreach News

A Thousand Words: Advanced Visualization for the Humanities

Purpose

Visualization uses computers to find patterns and make connections that normally cannot be seen. This project will develop the software tools, skills, and knowledge base to allow humanities researchers to use visualization - specifically on high-resolution displays powered by supercomputers – to perform novel research.



Overview

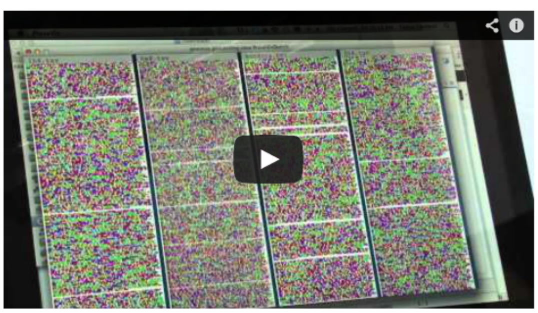
The tools that are currently available for visualization on high-resolution displays are primarily targeted at scientists, and because of that, the tools can be complicated and ill suited to the needs of humanities scholars. To address this issue, we have chosen a programming language called Processing as a platform on which to build visualizations and visualization tools in collaboration with humanities scholars.

Processing makes it easy to create simple interactive "sketches" that combine text, video, 3D graphics, animation, sound, and more. However, like most software, Processing is not currently capable of running on ultra-high-resolution display clusters. The results of this start up project will enable researchers anywhere to use Processing in conjunction with tiled displays at universities, museums and research centers.

Our long-term goal with this project is to create the world's most advanced platform for humanities visualization. We aim to create software tools that will enable a new class of scholars from the humanities to use high-resolution displays and advanced computing to create visualizations, interactive maps, and multimedia works at a scale and resolution never before possible.

Massive Pixel Environment (MPE)
We've developed a free and open-source software library called Massive Pixel Environment that makes it possible to render interactive Processing sketches across distributed computing systems on many displays.

More information and download link here.



Funding Source

National Endowment for the Humanities (NEH)

TACC Staff

Rob Turknett
Digital Media, Arts & Humanities Coordinator

Brandt Westing
Visualization Laboratory Manager
Research Engineering/Scientist Associate II

Partners

- Tanya Clement, UT Austin, School of Information
- Matt Cohen, UT Austin, Department of English
- Paul Resta, UT Austin, College of Education
- Jason Baldridge, UT Austin, Department of Linguistics

Research & Development
User Research
SciVis Gallery
Publications
TACC Projects
TACC Software

Related Links
National Endowment for the Humanities
Massive Pixel Environment (MPE)

Contact Us
Rob Turknett
Digital Media, Arts & Humanities Coordinator
turknett@tacc.utexas.edu

Google™ Custom Search | Search TACC

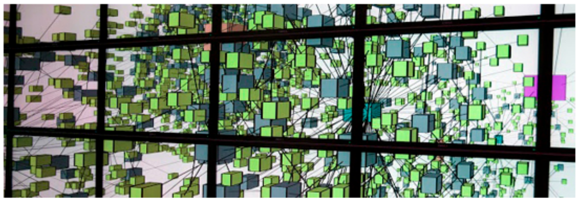
Office of the Vice President for Research | Feedback | Index | Facebook | Twitter | Contact

TACC TEXAS ADVANCED COMPUTING CENTER

Powering Discoveries That Change The World

THE UNIVERSITY OF TEXAS AT AUSTIN

Home | About | Resources | User Services | Research & Development | Partnerships | Education & Outreach | News



Massive Pixel Environment (MPE)

Purpose

Massive Pixel Environment is a library for extending Processing sketches to multi-node tiled displays. This library makes it possible to render interactive Processing sketches across distributed computing systems on many displays. It is intended for tiled display systems, but works in many other types of environments. With simple modifications, a sketch can be rendered across a cluster at the native resolution of the displays, and can greatly increase the amount of data that can be visualized at one time. This library is intended to run on Linux and OSX-based clusters.

This library is developed from scratch at the TACC/ACES Visualization Lab with inspiration from Most Pixels Ever, developed by Daniel Shiffman.

This work was made possible by funding from the National Endowment for the Humanities (NEH) Grant: HD-51475-11, A Thousand Words: Advanced Visualization for the Humanities.

Massive Pixel Environment features:

- Create interactive multimedia and data visualizations that span multiple displays, at a resolution and scale never before possible
- Enables extremely high resolution Processing sketches -- tested on the 328 Megapixel Stallion display cluster at the TACC/ACES Visualization Lab

The Massive Pixel Environment library is available for download on GitHub. Massive Pixel Environment is free software issued by the University of Texas at Austin under the BSD license.

TACC Staff

Brandt Westing
Visualization Laboratory Manager
Research Engineering/Scientist Associate II

Rob Turknett
Digital Media, Arts & Humanities Coordinator

Research & Development

User Research
SciVis Gallery
Publications
TACC Projects
TACC Software

Related Links

National Endowment for the Humanities
A Thousand Words: Advanced Visualization for the Humanities

Contact Us

Rob Turknett
Digital Media, Arts & Humanities Coordinator
turknett@tacc.utexas.edu

Google™ Custom Search | Search TACC

Office of the Vice President for Research | Feedback | Index | Facebook | Twitter | Contact

Appendix 4: Press Release

Webpage Screenshot

TACC TEXAS ADVANCED COMPUTING CENTER

Powering Discoveries That Change The World

THE UNIVERSITY OF TEXAS AT AUSTIN

Home | About | Resources | User Services | Research & Development | Partnerships | Education & Outreach | News

Most Pixels Ever: Cluster Edition

TACC Develops Visualization Software for Humanities Researchers

If you have trouble viewing this video, please visit TACC's YouTube page.

AUSTIN, Texas - The Texas Advanced Computing Center (TACC) at The University of Texas at Austin has released *Most Pixels Ever: Cluster Edition*, an open source software tool that allows researchers, especially those in the humanities, to create interactive, multimedia visualizations on high resolution, tiled displays like TACC's Stallion, one of the highest resolution tiled displays in the world at 328 million pixels.

"The goal is to make visualization tools easier for humanities researchers to use," said Rob Turknett, digital media, arts and humanities coordinator at TACC. "The proliferation of digitized textual, visual and aural resources is a great boon for the humanities, offering opportunities for new kinds of scholarship, but it also brings a new complexity."

Supported by a startup grant from the National Endowment for the Humanities titled "A Thousand Words: Advanced Visualization for the Humanities," the software is based on a language called Processing, a programming toolkit that makes it easier for people to create visualizations.

"As the amount of cultural data that scholars work with increases, it becomes crucial to visualize that data on a sufficiently high resolution display," Turknett continued. "Conventional display resolutions simply aren't keeping pace with this explosion of online cultural data to be explored."

The work borrows ideas from a library called *Most Pixels Ever* by Daniel Shiffman at the Interactive Telecommunications Program at NYU's Tisch School of the Arts. However, Shiffman's version required considerable configuration from users, according to Brandt Westing, technical lead on the project and manager of the TACC/ACES Visualization Lab (Vislab). "Using Shiffman's work as an inspiration, we re-wrote the software from scratch to work on any type of composite display from laptops to the highest-end visualization clusters and tiled displays."

Visualization clusters and tiled displays allow small groups of people to collaboratively explore large amounts of data and many types of visualizations, including: high resolution imagery (satellite, aerial photography, scientific instruments); high resolution movies (hi-res animations, time-series simulation results); 2D information display (maps, charts, graphs, data, text); and 3D visualization (complex geometries, interactive exploration of 3D datasets).

"Most of the tools that exist for these displays are developed by and for scientists, yet there are many researchers from the humanities and arts who want to do visualization," Turknett said. "The software that we've developed is part of an effort to make advanced visualization systems more accessible to people who may not have a deep technical background."

Jason Baldridge, an associate professor in the Linguistics Department at The University of Texas at Austin, researches a wide range of problems involving the connection among language, computation, geography and time. His research has the potential to improve a variety of applications based on natural language processing and text analytics that are widely used to analyze unstructured data.

"We're awash in very large collections of text and we simply cannot read through all of them," Baldridge said. "We need improved tools for exploring text collections so people can find interesting patterns, and this new software developed by TACC can help us accomplish this goal."

Baldridge's current project involves analyzing a collection of several hundred texts from the Civil War. "Using the new software on TACC's Stallion, we're parallelizing the computations to do visualizations and view an enormous amount of data at once, both of which are incredibly useful in exploring the output from our models and applications." For example, Baldridge uses the software to identify text passages from memoirs that are connected to a particular city and time.

"And, because they connect language to the real world, they lend themselves to novel visualizations that illustrate the geographical and historical context of text collections and language use," Baldridge said.

Tanya Clement, an assistant professor at the School of Information, builds tools for scholars who analyze literary texts. "Humanities researchers have not had access to large data sets until recent decades. It's essential for humanities scholars to be involved in the creation of new software and tools so the concerns of the community are reflected," Clement said.

Both Baldridge and Clement collaborated with TACC on the project.

Most Pixels Ever: Cluster Edition is already in use at two other institutions: The University of Texas at El Paso and The University of Texas at San Antonio's Center for Simulation Visualization and Real-time Prediction.

Most Pixels Ever: Cluster Edition is open source and available for download. For more information, visit Most Pixels Ever: Cluster Edition and A Thousand Words: Advanced Visualization for the Humanities.

###

Date Posted: 2012-12-11   *Faith Singer-Villalobos*

✓ to-do | Share |

Connect

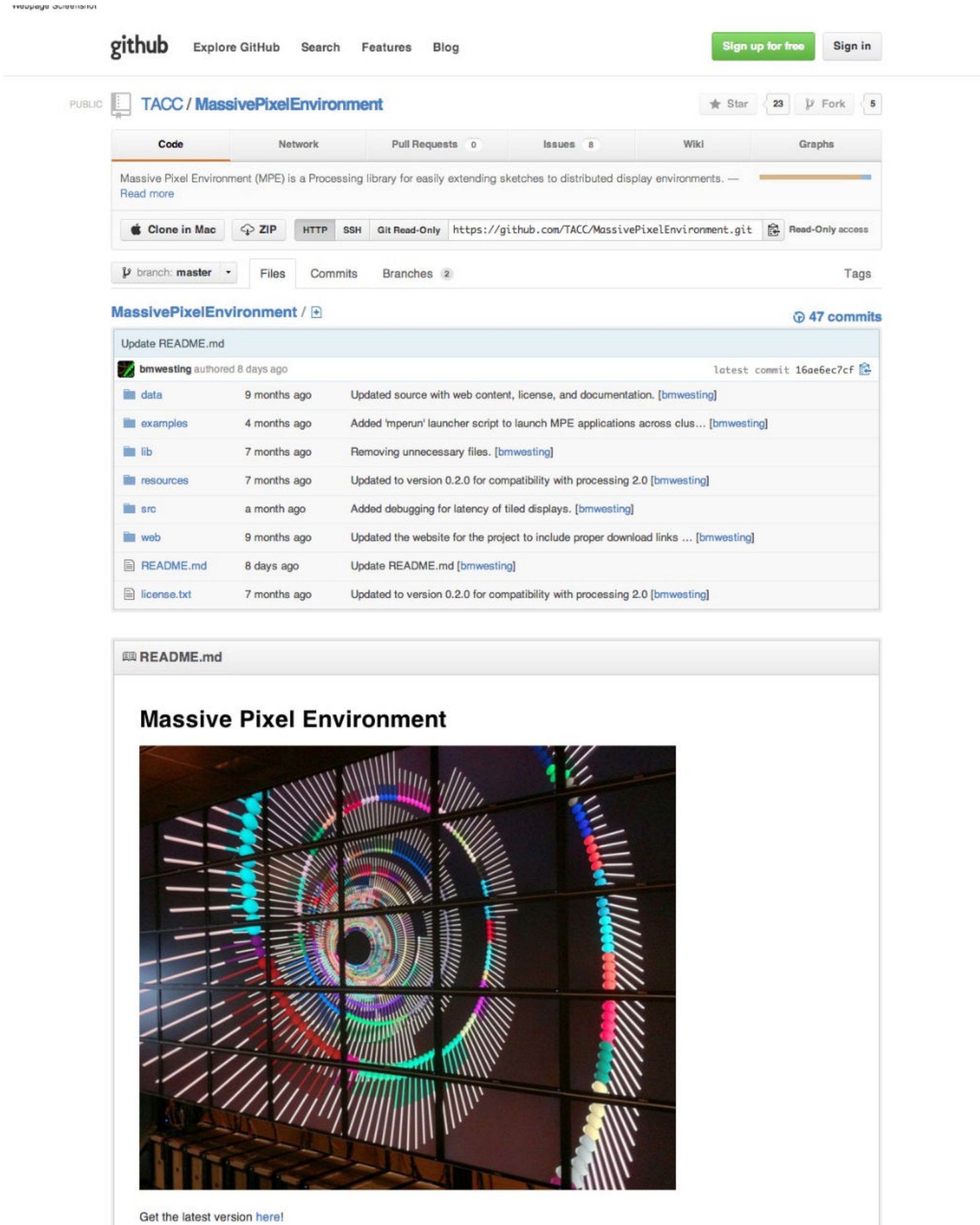
Quick Links
Feature Stories
Multimedia
Events

Press Releases Archive
2012 Press Releases
2011 Press Releases
2010 Press Releases
2009 Press Releases

Contact
Faith Singer-Villalobos
Public Relations
faith@tacc.utexas.edu

http://www.tacc.utexas.edu/news/press-releases/2012/tacc-develops-visualization-software

Appendix 5: GitHub Page



Massive Pixel Environment (MPE) is a Processing library for easily extending sketches to distributed display environments. MPE was developed at the Texas Advanced Computing Center and aims to make it easy to prototype visualizations for otherwise difficult to use distributed systems. MPE is compatible with the lastest Processing versions and supports full compatibility with advanced featuresets of OpenGL.

In MPE:

- Each node in a cluster runs an instance of the Processing sketch application
- The Processing sketch is easily launched across the cluster through the use of a single configuration file
- Support for all features of Processing is available

Getting Started:

- MPE Simple Guide

Other Links

- MPE: CE Project Page
- MPE: CE Paper/Abstract

Credits

NATIONAL ENDOWMENT FOR THE Humanities

This work was made possible by funding from the National Endowment for the Humanities (NEH) Grant:HD-51475-11, A Thousand Words: Advanced Visualization for the Humanities.

Awards and News

DH Awards 2012 Winner: Best DH visualization or infographic

Appendix 6: GitHub Tutorial (page 1)

Webpage Screenshot

github Explore GitHub Search Features Blog Sign up for free Sign in

PUBLIC TACC / MassivePixelEnvironment Star 23 Fork 5

Code | Network | Pull Requests 0 | Issues 8 | Wiki | Graphs

Home Pages Wiki History Git Access

MassivePixelEnvironment HowTo

Page History

Massive Pixel Environment (MPE) is inspired by the MostPixelsEver library that written for Processing a few years ago. Massive Pixel Environment is written from scratch to better support simple configuration, ease-of-use, and speed in environments with many nodes and displays when compared to MostPixelsEver.

Before you get started, you need to make sure you have MPE downloaded. You can get it from the project page here. You may also need to download the PeasyCam library for this example, if you don't have it already.

Getting started with your first MPE program:

1. Creating a configuration file for your display.
2. Writing your first MPE program.
3. Running your first MPE program.

Read the FAQ for common problems/answers.

Last edited by bmwesting, 8 days ago

GitHub	Applications	Services	Documentation	More
About us	GitHub for Mac	Gauges: Web analytics	GitHub Help	Training
Blog	GitHub for Windows	Speaker Deck: Presentations	Developer API	Students & teachers
Contact & support	GitHub for Eclipse	Gist: Code snippets	GitHub Flavored Markdown	The Shop
GitHub Enterprise	GitHub mobile apps	Job board	GitHub Pages	Plans & pricing
Site status				The Octodex

Terms of Service Privacy Security

https://github.com/TACC/MassivePixelEnvironment/wiki/MassivePixelEnvironment-HowTo